Pattern Recognition, 2000. Proceedings. 15th International Conference on
Volume 1, 3-7 Sept. 2000 Page(s):644 - 647 vol.1

Progress in Automated Evaluation of Curved Surface Range Image Segmentation*

Jaesik Min, Mark W. Powell, and Kevin W. Bowyer
Department of Computer Science and Engineering
University of South Florida
Tampa, FL 33620
jmin, mpowell, kwb@csee.usf.edu

Abstract

We have developed an automated framework for performance evaluation of curved-surface range image segmentation algorithms. Enhancements over our previous work include automated training of parameter values, correcting the artifact problem in K^2T scanner images, and acquisition of images of the same scenes from different range scanners. The image dataset includes planar, spherical, cylindrical, conical, and toroidal surfaces. We have evaluated the automated parameter tuning technique and found that it compares favorably with manual parameter tuning. We present initial results from comparing curved-surface segmenters by Besl and Jain and by Jiang and Bunke.

1. Introduction

Our initial work in performance evaluation of range image segmentation algorithms resulted in a comparison of four algorithms [2]. Two limitations of this study are: (1) the comparison was limited to algorithms that segment images into planar regions, and (2) the "training" to select parameter values for the segmenter was done manually, and differently, for each segmenter. Later work extended the evaluation to include algorithms that segment range images into curved-surface patches [5]. However, the parameter training process was still manual. A current effort evaluates additional planar-surface segmentation algorithms in the same manner as the original study [3]. This paper extends our previous work in several directions. Importantly, we introduce an automated method of training to select parameter values for the segmenter. We also correct the artifact problem discovered in the K^2T images in [5]. And we present results of comparing two segmenters [1, 4] on images from two different types of range scanners. We propose that competing segmenters can be fairly compared by automated training on a standard public data set, followed by testing on a sequestered data set.

*This work was supported by National Science Foundation grants IIS-9731821 and EIA-9729904.

2. Framework

Our definition of segmentation for range images is given in detail in [2] and is summarized here. For any image R consisting of $r_1 \ldots r_n$ subregions:

1. Every pixel belongs to a region.
2. Every region is four-connected.
3. All regions are disjoint.
4. All pixels in a region belong to the same surface.
5. If two regions are four-connected and adjacent then they represent different surfaces.
6. There are noise regions in the image where no valid measurement was possible which all have the same label (violating rule 2) and for which rules 4 and 5 do not apply.

We manually record "ground truth" for all images in our dataset using an interactive tool that we developed. To compare the performance of segmenters according to the ground truth, we divide the image dataset into a train set for parameter tuning and test set for evaluation. Image data sets from each different range scanner are evaluated independently. Both training and testing use performance metrics (as defined in [2]) for correctly detected, over- and under-segmented, missed and noise regions. These definitions are based on the degree of mutual overlap required between a region in the segmentation result and a region in the ground truth. Requiring a larger percentage of mutual overlap gives a stricter definition of correct segmentation. An automated comparison tool is used to compare the segmentation result to the ground truth over a range of values for the percentage of mutual overlap. This gives quantitative results for number of correctly segmented regions over the range of overlap thresholds, as in Figures 1 and 3.

3. Automated Training Procedure

A major improvement in our comparison framework is the addition of an automated training procedure. Manual parameter tuning means that training results vary according to the effort and skill of the experimenter, and so results may be hard to reproduce across experimenters. We developed an automated procedure for parameter training so that training results will be reproducible across experimenters. In comparing to results in [2], automated training produces results that are basically equivalent to manual training.

The number of parameters varies between segmenters, and is a critical factor in the execution time of a parameter search procedure. The number of executions of the segmenter on the initial training image is N^D, where N is the number of initial sample values of each parameter and D is the number of parameters. Initial experiments showed that $N = 5$ allowed the search process to find as good a parameter setting as larger N. For the segmenters that we have used, and the number of training images, it is practical to train as many as the 4 most important parameters of the segmenter. To keep the training computationally feasible, less significant parameters are fixed at default values.

The outline of the automated training is as follows:

```
Initial Step:
  Use I1 = 5x5x5x5 parameter sampling
           on training image 1.
  Take best I2 = 1/2 I1 of parameter
           points on to image 2.
  Take best I3 = 2/3 I2 of parameter
           points on to image 3.
  Take best I4 = 3/4 I3 of parameter
           points on to image 4.
  ... until all train images analyzed.

Successive Steps:
  Use I1 = 3x3x3x3 sampling around best
            current points on image 1.
  Take best I2 = 1/2 I1 of parameter
            points on to image 2.
  Take best I3 = 2/3 I2 of parameter
            points on to image 3.
  Take best I4 = 3/4 I3 of parameter
            points on to image 4.
  ... until all train images analyzed.

Until improvement between steps is < 5%.

Select best-performing parameter setting.
```

The initial step starts with an uniform sampling of the segmenter's parameter space. These parameter settings are run on the first training image. The better-performing 1/2 of these settings are run on the second training image. Then, the better-performing 2/3 of the parameter settings are run on the third training image. This continues until all of the training images are analyzed. Successive iterations begin with the better-performing points from the previous iteration and sample at the mid-points between previous neighboring points. In this way, the resolution of the sampling becomes finer at each iteration. When the improvement in performance between iterations falls below 5%, the parameter setting that gives best performance is selected as the trained parameter setting.

Table 1. Manual and automated training results for UB planar-surface segmenter (tuned on four most important parameters)

	ABW		Perceptron	
Para.	manual	automated	manual	automated
T_1	1.25	1.20	1.75	2.75
T_2	2.25	2.90	3.25	3.50
t_1	4.0	4.50	4.0	1.70
t_2	0.1	0.75	0.1	0.125
t_3	3.0	3.0	3.0	3.0
t_4	0.1	0.1	0.2	0.2
t_5	100	100	150	150

4. Automated Versus Manual Training

We compared automated parameter training with manual tuning previously performed by others using the same segmenter, train set and test set. Hoover et al. [2] performed an evaluation of planar-surface segmenters with ABW and Perceptron range images. For each image type, the train set consists of 10 images and test set 30 images. In [2], all of the segmenters were tuned manually by the people who developed the segmenter. They found that the University of Bern (UB) planar-surface segmenter offered good performance and was the fastest. The UB planar-surface segmenter has 7 input parameters, as shown in Table 1. The first two parameters are the most critical. The Table lists the values of the seven parameters found by the manual training in [2], and values of the first four parameters as found by our automated training method. The parameter values selected by automated training are generally close to those selected by manual training, and result in similar performance on the test set. Figure 1 shows the performance results of manual and automatically trained parameters on the two 30-image test sets used in [2]. The automatically trained UB segmenter shows slightly better performance on the Perceptron images and slightly worse performance on the ABW. Differences in performance here due to manual versus automated training are small in comparison to dif-

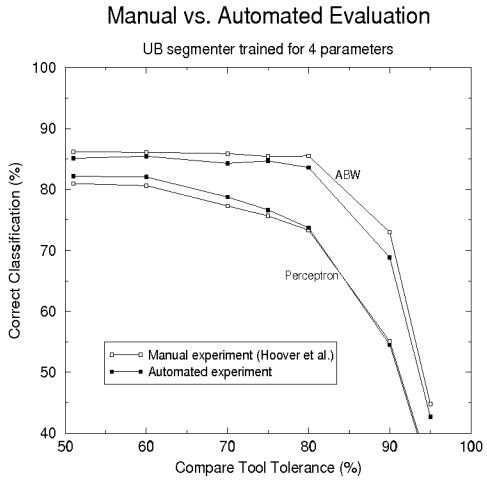


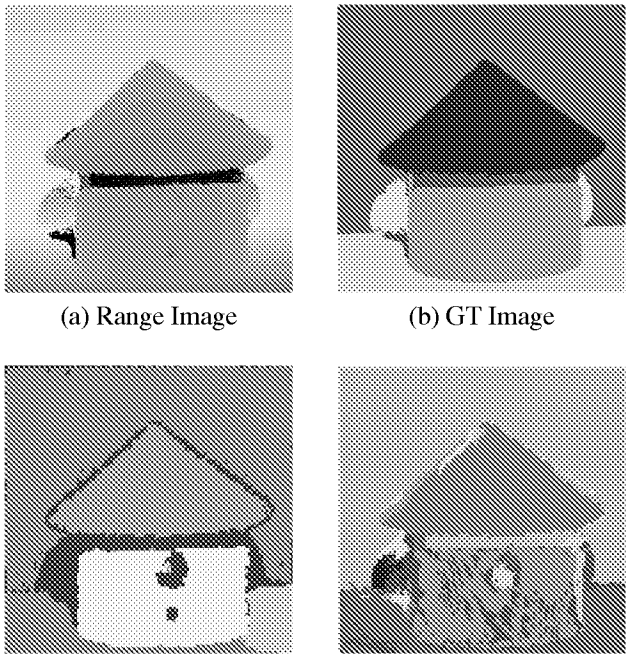
Figure 1. UB planar-surface test results, trained manually and automatically.

ferences between segmenters [2]. This indicates that automated training provides results comparable to those from manual training by the developer of the algorithm.

5. Curved-Surface Segmenter Evaluation

We have acquired a train set of 256x256 images from 15 scenes imaged with both the K^2T structured light scanner and the Cyberware 3030HRC range finder. The K^2T images are of higher quality than those used in [5] due to a corrected interpolation procedure used to estimate the 3D positions of points that do not lie on the edge of a projected light stripe. Figure 2 (a) and (b) show one of the train images and its manually-created GT image.

We have used our automated training procedure to train two curved-surface segmentation algorithms [1, 4] using this set of images. Table 2 is a list of parameters tuned by automated training and Figure 2 (c) and (d) show sample machine segmentations from both segmenters. The overall results of this training evaluation are shown in Figure 3. The results suggest that the UB curved-surface segmentation algorithm substantially out-performs the Besl and Jain algorithm. However, these results should be interpreted carefully. The Besl and Jain algorithm performs variable-order surface fitting, whereas the UB algorithm considers only a pre-defined set of quadric surfaces (e.g., sphere, cone, cylinder, ...). This difference also results in the UB algorithm executing much faster than the Besl and Jain algorithm. Execution time for either algorithm varies with scene content, but the UB segmenter might take 10 seconds to segment a "typical" 256x256 image on a SUN Sparc Ultra 5, whereas the Besl and Jain would take several minutes.



(a) Range Image (b) GT Image

(c) UB segmentation (d) Besl & Jain Segmentation

Figure 2. Sample machine segmentation result on a K^2T range image

6. Discussion

We have outlined basic elements of an automated framework for objectively evaluating curved-surface range segmentation algorithms. The framework includes automated parameter tuning. Evidence shows that automated parameter tuning gives results that are comparable to careful and knowledgeable manual tuning.

We compared two curved-surface range segmentation algorithms. The Jiang and Bunke algorithm gives a higher rate of correctly segmented regions and has a much faster execution time. The performance difference is larger for images from the K^2T structured-light scanner than it is for images from the Cyberware. Since both scanners were used to image the same set of scenes, this indicates that the properties of the Cyberware images are somehow more suitable to the Besl and Jain algorithm than are the properties of the K^2T images. Based on this performance assessment, the UB algorithm seems a good choice for use in future performance comparisons. For algorithms with execution speed comparable to the UB algorithm, the search of the parameter space in the training phase can be made more exhaustive and less heuristic.

We envision a general, automated performance evaluation framework for range segmenters (planar- or curved-

surface) that would work as follows. The framework would consist of an algorithm development package, and a performance benchmark step. The algorithm development package would consist of multiple sets of training images, ground truth for the training images, a utility for scoring segmentation results compared to ground truth, a utility for automated parameter training, and a baseline comparison algorithm (e.g., the UB algorithm). Algorithm developers would be able to compare the performance of their proposed algorithm, in an automated and objective fashion, to that of the baseline algorithm on different (training) image sets. The complete algorithm development package would be publicly available from the USF Computer Vision Lab web site.

The performance benchmark step would come after algorithm development reaches a stable state, presumably at the point where the candidate algorithm out-performs the baseline algorithm on the training data. The input to the performance benchmark step would be the code for the candidate algorithm, and a list of the parameter values found for each training set. The candidate algorithm would then be used to segment a separate test set of images, one time for the parameter values found on each training set. The test set would be sequestered (not publicly available). This would result in a set of performance curves such as those in Figure 3. The performance of the candidate algorithm could be compared to that of the baseline algorithm in a quantitative manner, based on area under the performance curves, with a test for statistical significance of the results. We are currently working toward a complete implementation of this framework. It is hoped that such a framework would promote objective comparison of algorithm performance, which in turn will stimulate research focused on development of better-performing algorithms.

References

[1] P. Besl and R. Jain. Segmentation through variable-order surface fitting. *IEEE PAMI*, 10(2):167–192, March 1988.

[2] A. Hoover, G. Jean-Baptiste, and et al. An experimental comparison of range image segmentation algorithms. *IEEE PAMI*, 18(7):673–689, July 1996.

[3] X. Jiang, K. Bowyer, and et al. Some further results of experimental comparison of range image segmentation algorithms. In *International Conference on Pattern Recognition*, page (this proceedings), Barcelona, 2000.

[4] X. Jiang and H. Bunke. Range image segmentation: Adaptive grouping of edges into regions. In *Asian Conference on Computer Vision*, Hong Kong, 1998.

[5] M. W. Powell, K. W. Bowyer, X. Jiang, and H. Bunke. Comparing curved-surface range image segmenters. In *International Conference on Computer Vision*, Mumbai, India, January 1998.

Table 2. Training result of curved-surface segmenter. Four parameters are tuned.

Parameter	K^2T	Cyberware
UB curved-surface segmenter		
T_g: Segment Tolerance	0.19	0.73
T_r^p: RMS Error (Plane)	0.901	1.0
T_r^c: RMS Error (Curve)	0.901	0.01
T_s: Minimum Region Size	300	260
T_j: Jump Edge	1.0	1.0
T_c: Crease Edge	30.0	30.0
T_a^p: Average Fit Error (Plane)	0.07	0.07
T_a^c: Average Fit Error (Curve)	0.09	0.09
T_f^p: Tolerable Fit Error (Plane)	2	2
T_f^c: Tolerable Fit Error (Curve)	3	3
Besl & Jain segmenter		
RMS error limit	3550	2170
Error limit factor for growth	215	245
Pre-smooth window size	13	13
Derivative window size	13	9
HK-smooth window size	5	5
Min. initial region size	30	30
Max. number of iterations	30	30
Acceptability error factor	150	150
Max. blackboard region size	50	50
Regions test threshold	20	20

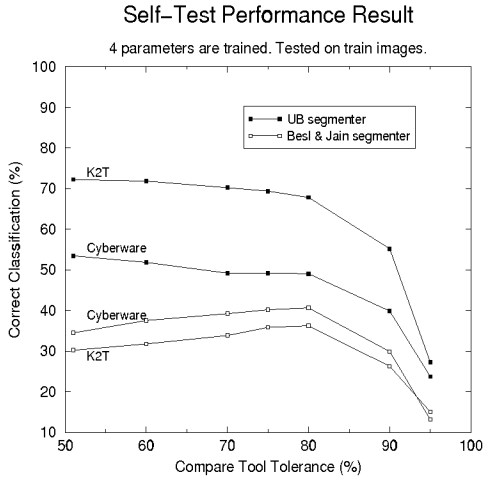


Figure 3. Comparison of two segmenters